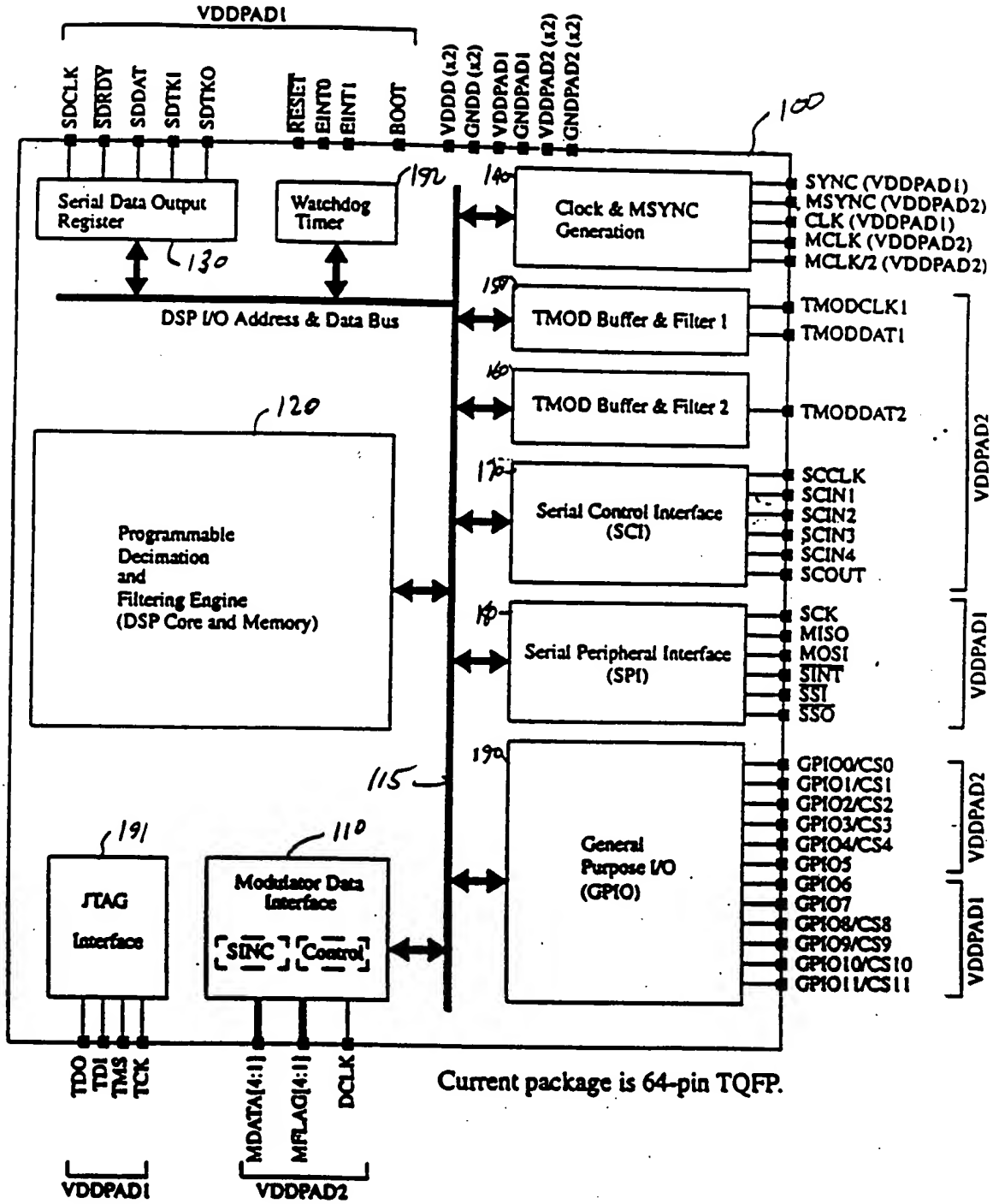


Current package is 64-pin TQFP.

FIGURE 1

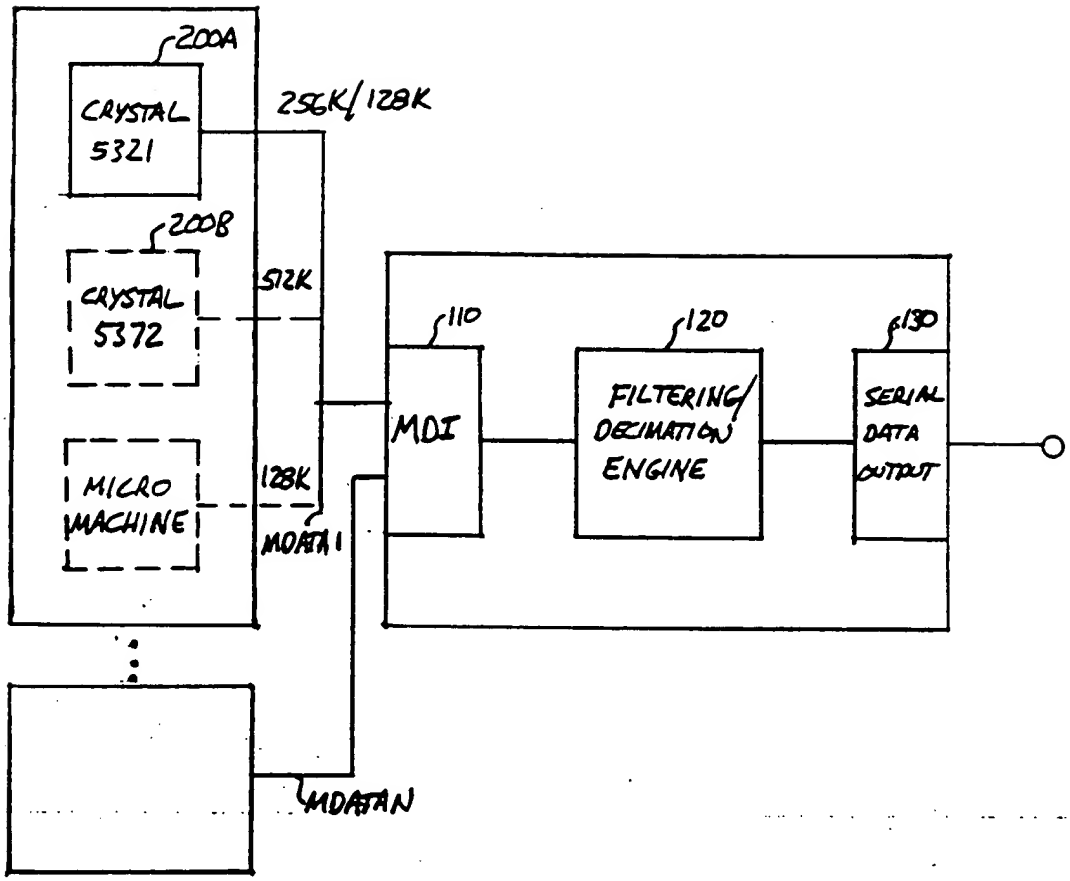

200A
CRYSTAL 5321
256K/128K
200B
CRYSTAL 5372
512K
MICRO MACHINE
128K
MDATA1
MDATAN
110
MDI
120
FILTERING/ DECIMATION ENGINE
130
SERIAL DATA OUTPUT

FIGURE 2

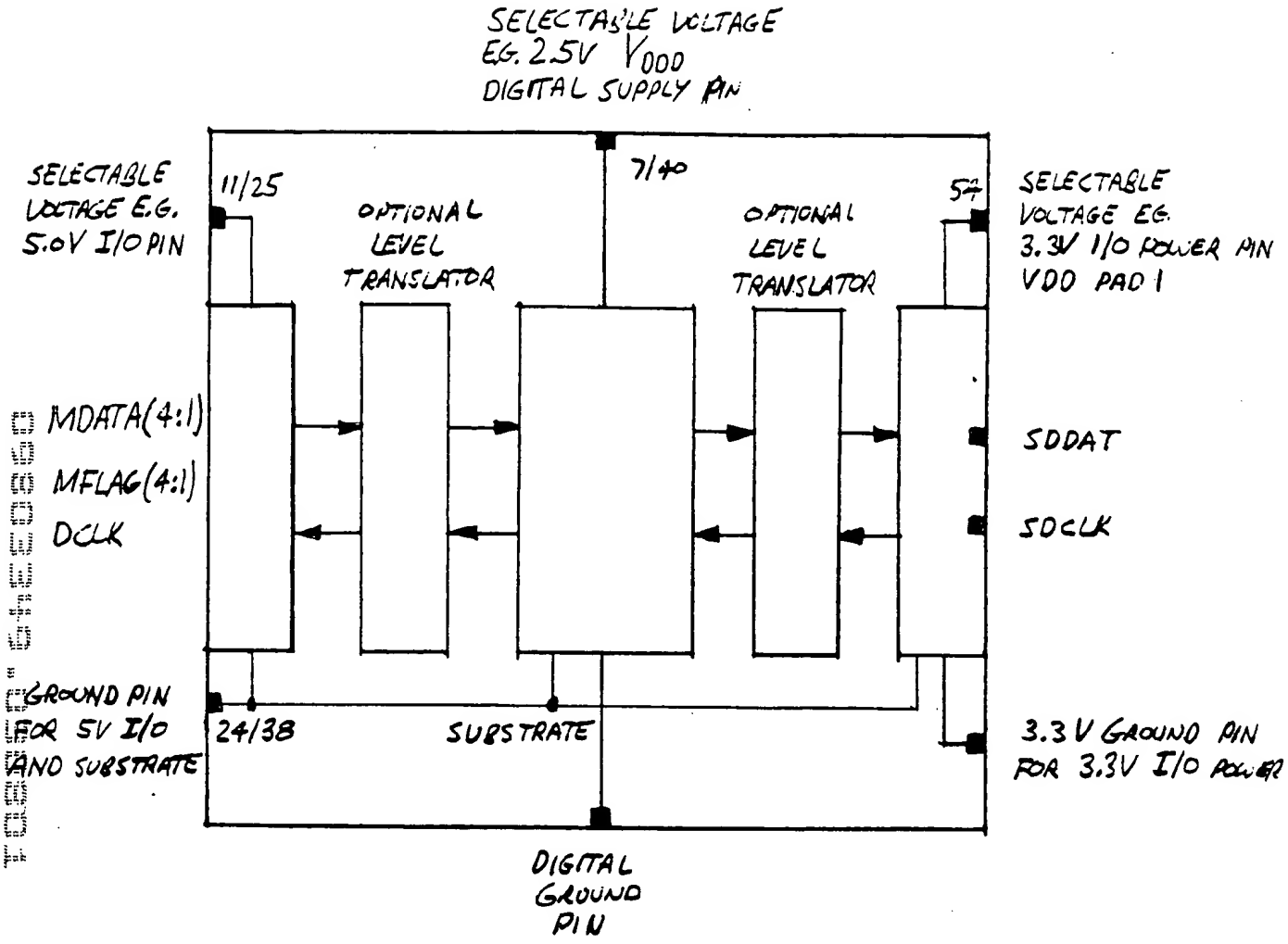

SELECTABLE VOLTAGE
EG. 2.5V VDDD
DIGITAL SUPPLY PIN
SELECTABLE VOLTAGE E.G. 5.0V I/O PIN
11/25
OPTIONAL LEVEL TRANSLATOR
7/40
OPTIONAL LEVEL TRANSLATOR
54
SELECTABLE VOLTAGE EG. 3.3V I/O POWER PIN VDD PAD I
MDATA(4:1)
MFLAG(4:1)
DCLK
SDDAT
SDCLK
GROUND PIN FOR 5V I/O AND SUBSTRATE
24/38
SUBSTRATE
3.3V GROUND PIN FOR 3.3V I/O POWER
DIGITAL GROUND PIN

FIGURE 3

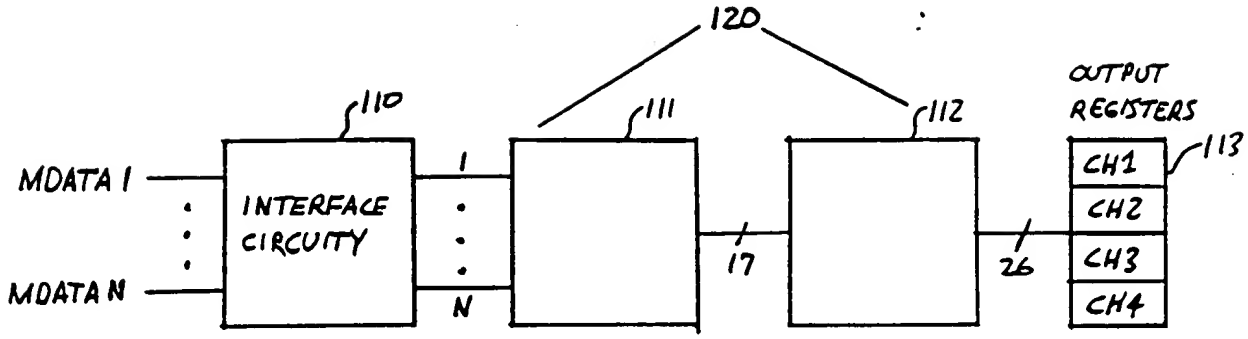

120
110
111
112
OUTPUT
REGISTERS
113
MDATA 1
MDATA N
INTERFACE
CIRCUITY
1
N
17
26
CH1
CH2
CH3
CH4

FIGURE 4

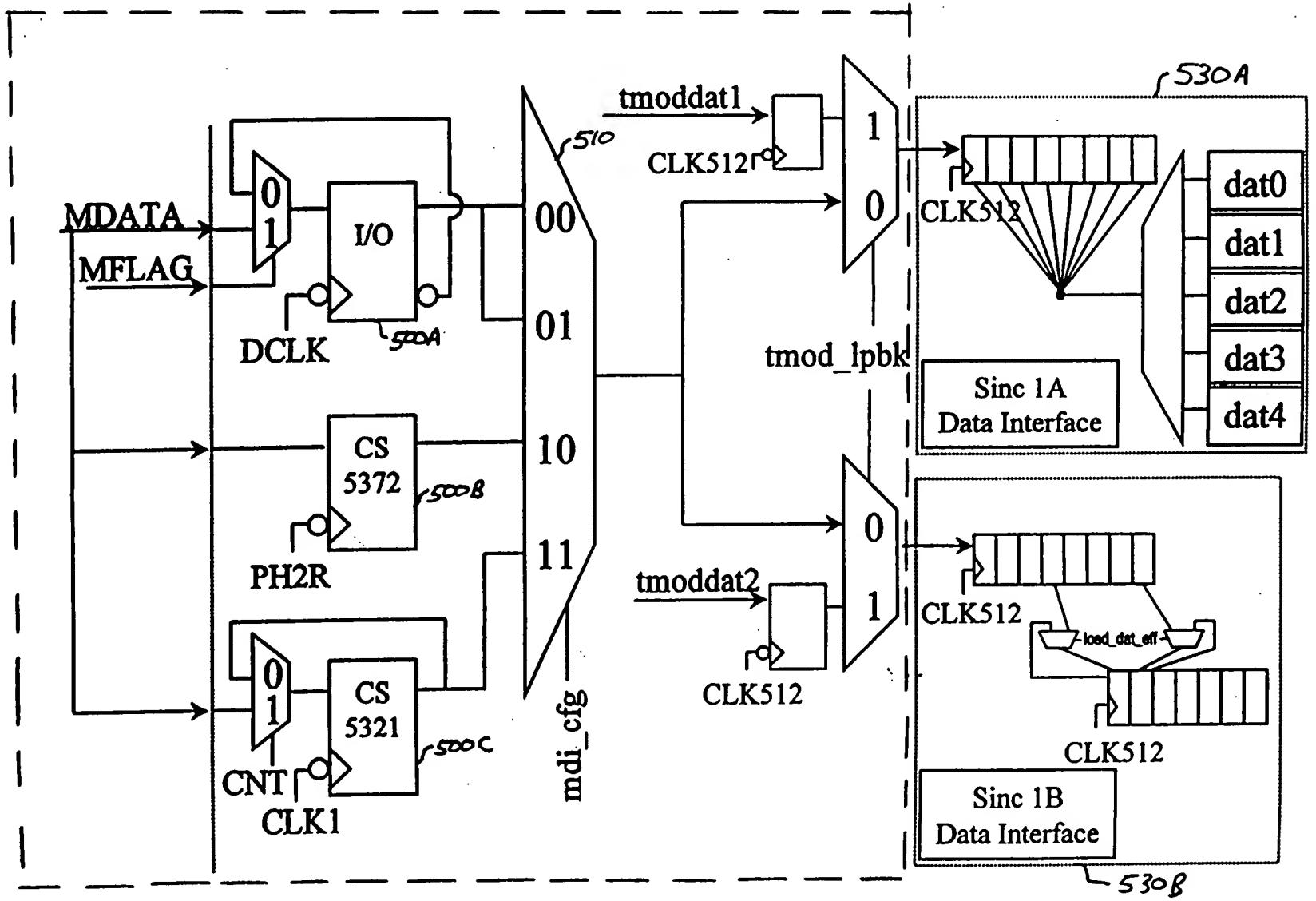


FIGURE 5

2. Bismarck Sinc Decimation Chain

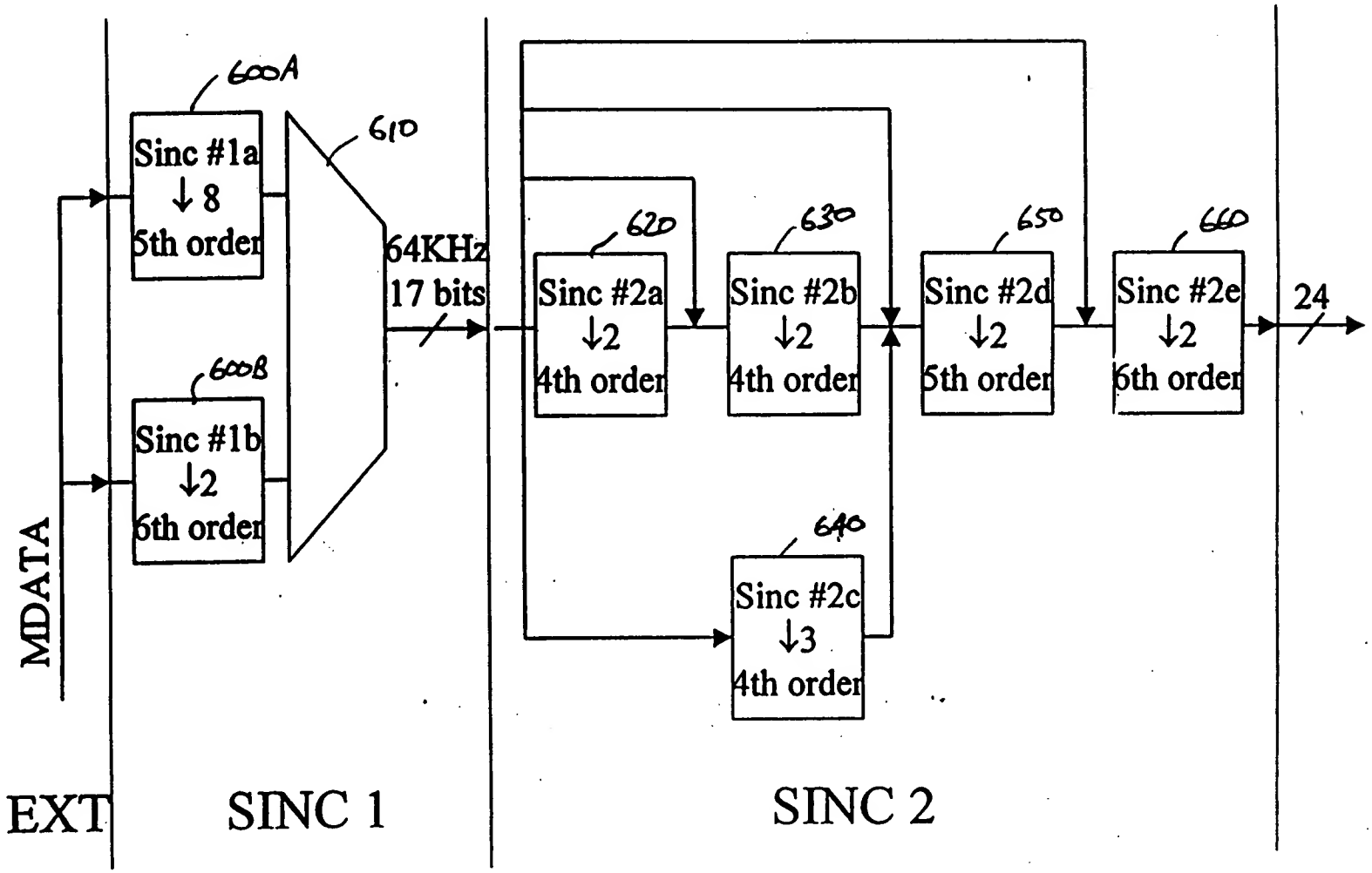


FIGURE 6

- Fifth order decimate by 8:

$$H(z) = \left(\frac{1 - z^{-8}}{1 - z^{-1}}\right)^5$$

- 36 tap FIR filter. Half of the (symmetric) coefficients

h_0=1	h_1=5	h_2=15	h_3=35	h_4=70	h_5=126	h_6=210	h_7=330	h_8=490
h_9=690	h_{10}=926	h_{11}=1190	h_{12}=1470	h_{13}=1750	h_{14}=2010	h_{15}=2226	h_{16}=2380	h_{17}=2460

FIGURE 7

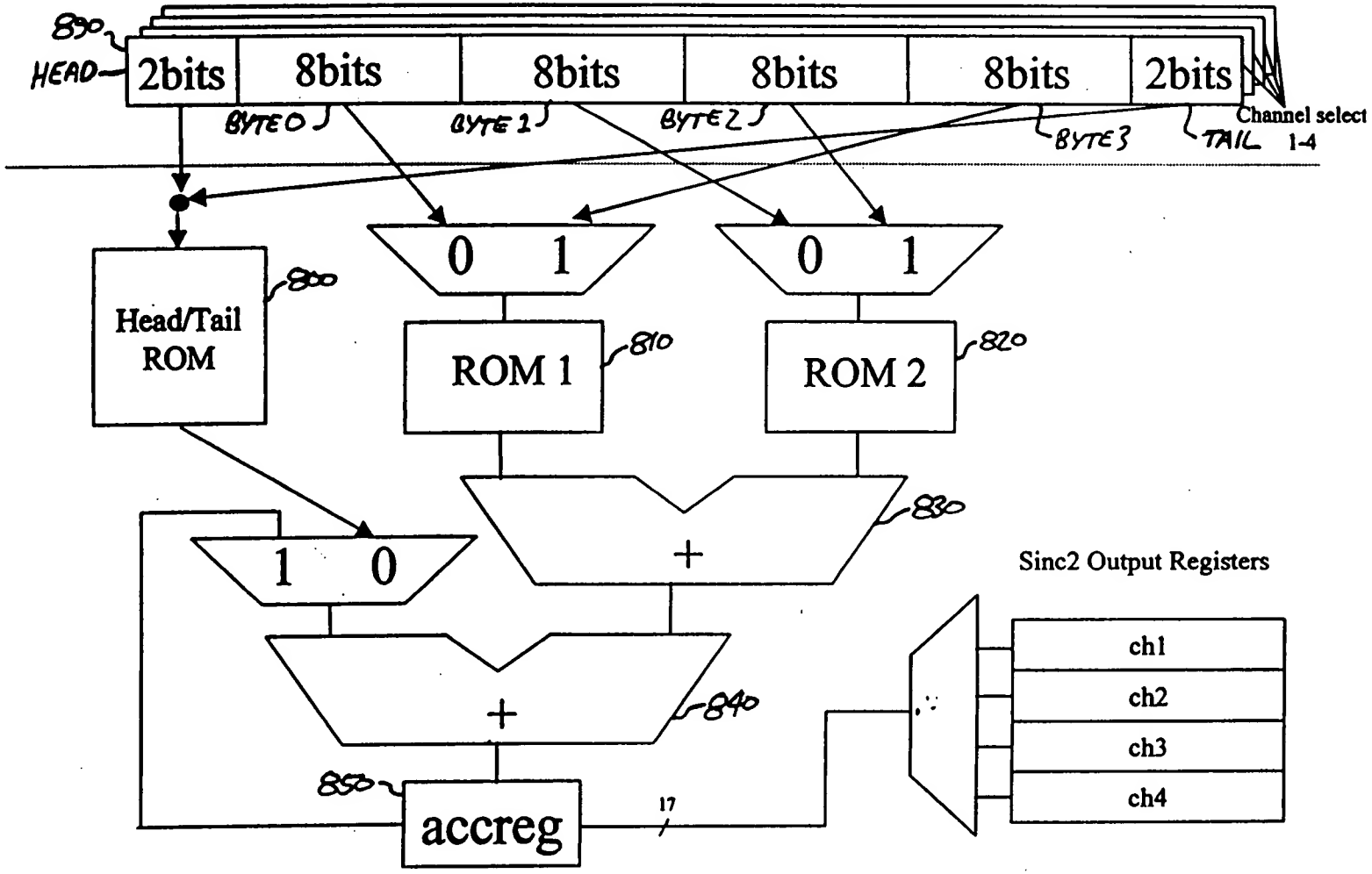

890
HEAD
2bits
8bits
8bits
8bits
8bits
2bits
BYTE 0
BYTE 1
BYTE 2
BYTE 3
TAIL
Channel select 1-4
0 1
0 1
Head/Tail ROM
800
ROM 1
810
ROM 2
820
830
+
1 0
Sinc2 Output Registers
ch1
ch2
ch3
ch4
+
840
850
accreg
17

FIGURE 8

$$H(z) = \left(\frac{1 - z^{-2}}{1 - z^{-1}}\right)^6$$

Impulse Response:

$$y[n] = x[n] + 6 \cdot x[n-1] + 15 \cdot x[n-2] + 20 \cdot x[n-3] + 15 \cdot x[n-4] + 6 \cdot x[n-5] + x[n-6]$$

FIGURE 9

3. Bismarck Sinc1b Functional Diagram

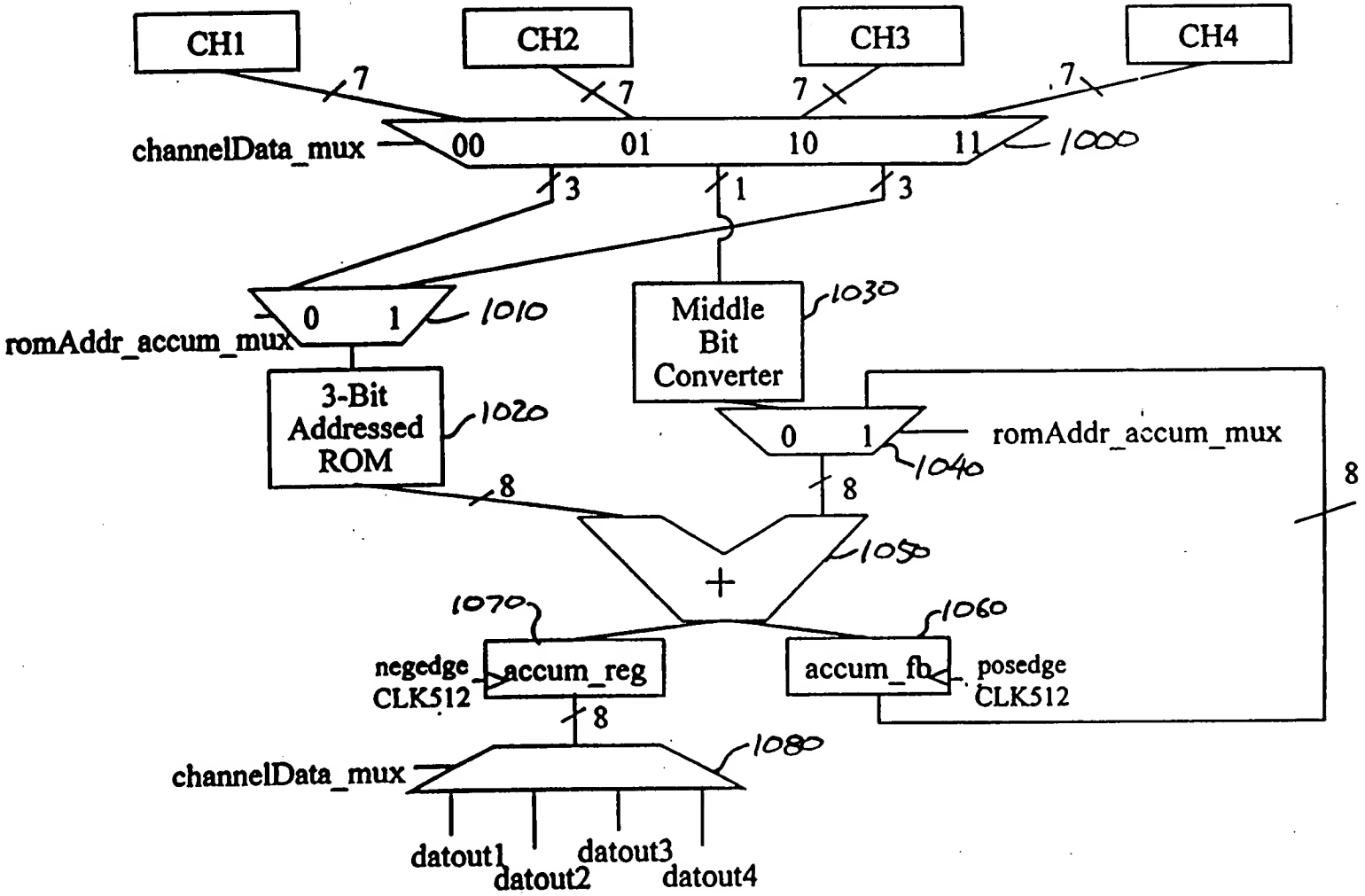


FIGURE 10

Filter Name	System Function	IMPULSE RESPONSE (FILTER COEFFICIENTS)
Sinc2(a) Sinc2(b)	$H(z)=\left(\frac{1-z^{-2}}{1-z^{-1}}\right)^4$	h[n] = [1 4 6 4 1]
Sinc2(c)	$H(z)=\left(\frac{1-z^{-3}}{1-z^{-1}}\right)^4$	h[n] = [1 4 10 16 19 16 10 4 1]
Sinc2(d)	$H(z)=\left(\frac{1-z^{-2}}{1-z^{-1}}\right)^5$	h[n] = [1 5 10 10 5 1]
Sinc2(e)	$H(z)=\left(\frac{1-z^{-2}}{1-z^{-1}}\right)^6$	h[n] = [1 6 15 20 15 6 1]

FIGURE 11

Sinc2(a) and Sinc2(b):

$$y[n] = x[n] + 4x[n-1] + 6x[n-2] + 4x[n-3] + x[n-4]$$
$$= x[n] + 4x[n-1] + 4x[n-2] + 2x[n-2] + 4x[n-3] + x[n-4]$$

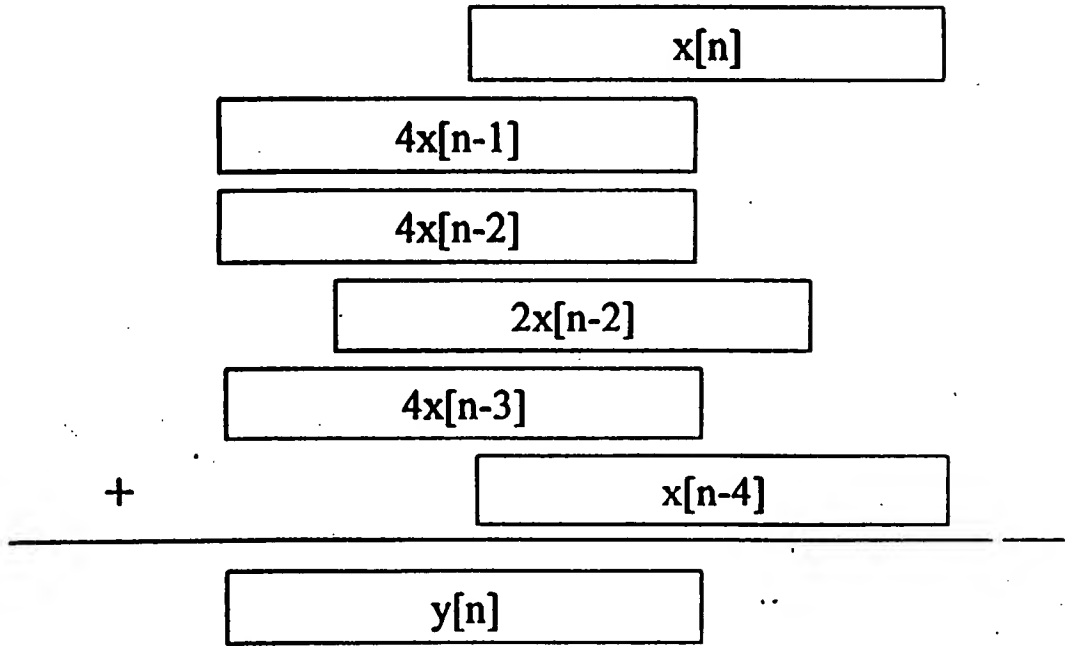


FIGURE 12

FIGURE 13A **Sinc2(c):**

$$y[n] = x[n] + 4x[n-1] + 10x[n-2] + 16x[n-3] + 19x[n-4] + 16x[n-5] + 10x[n-6] + 4x[n-7] + x[n-8]$$

$$= x[n] + 4x[n-1] + \boxed{8x[n-2] + 2x[n-2]} + 16x[n-3] + \boxed{16x[n-4] + 2x[n-4] + x[n-4]}$$

$$+ 16x[n-5] + \boxed{8x[n-6] + 2x[n-6]} + 4x[n-7] + x[n-8]$$

FIGURE 13B **Sinc2(d):**

$$y[n] = x[n] + 5x[n-1] + 10x[n-2] + 10x[n-3] + 5x[n-4] + x[n-5]$$

$$= x[n] + \boxed{4x[n-1] + x[n-1]} + \boxed{8x[n-2] + 2x[n-2]} + \boxed{8x[n-3] + 2x[n-3]} + \boxed{4x[n-4] + x[n-4]} + x[n-5]$$

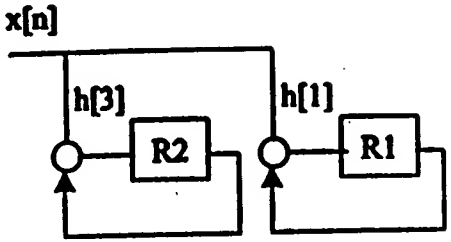
FIGURE 13C **Sinc2(e):**

$$y[n] = x[n] + 6x[n-1] + 15x[n-2] + 20x[n-3] + 15x[n-4] + 6x[n-5] + x[n-6]$$

$$= x[n] + \boxed{4x[n-1] + 2x[n-1]} + \boxed{16x[n-2] - x[n-2]} + \boxed{16x[n-3] + 4x[n-3]}$$

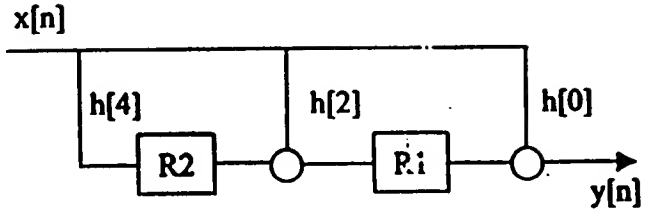
$$+ \boxed{16x[n-4] - x[n-4]} + \boxed{4x[n-5] + 2x[n-5]} + x[n-6]$$

Sinc2(a) and Sinc2(b):



Accumulate Phase (2 additions)

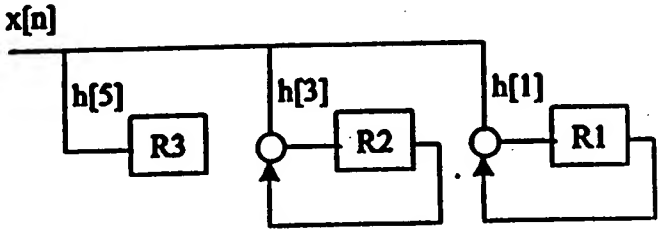
FIGURE 14A



Output Phase (4 additions)

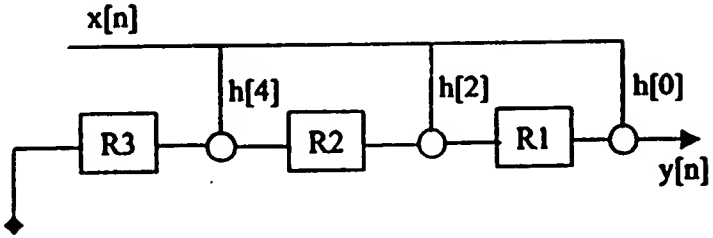
FIGURE 14B

Sinc2(d):



Accumulate Phase (5 additions)

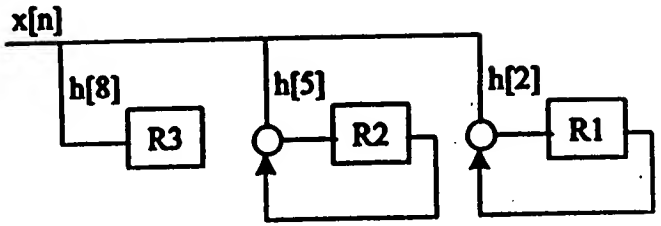
FIGURE 15A



Output Phase (5 additions)

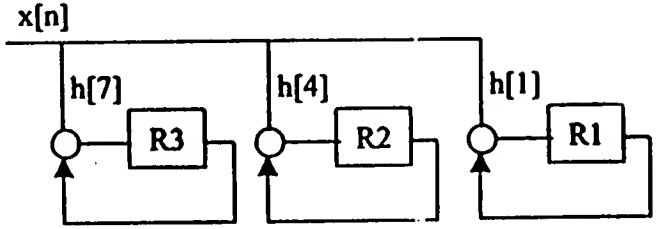
FIGURE 15B

Sinc2(c):



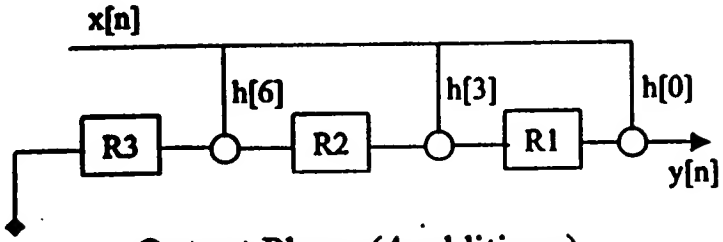
Accumulate Phase A (4 additions)

FIGURE 16A



Accumulate Phase B (5 additions)

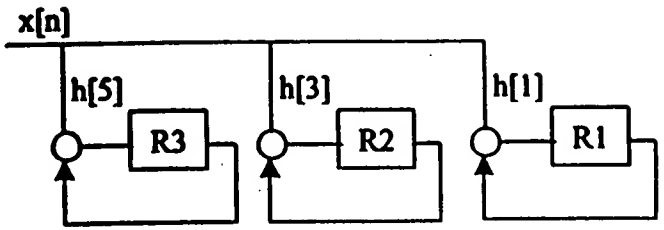
FIGURE 16B



Output Phase (4 additions)

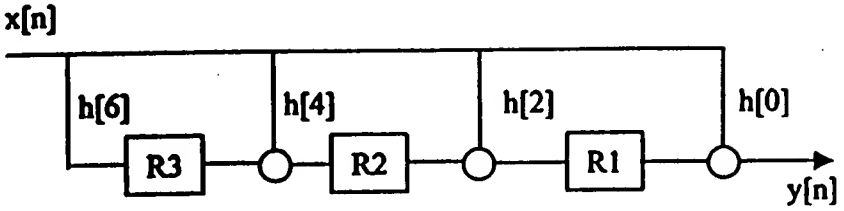
FIGURE 16C

Sinc2(e):



Accumulate Phase (6 additions)

FIGURE 17A



Output Phase (6 additions)

FIGURE 17B

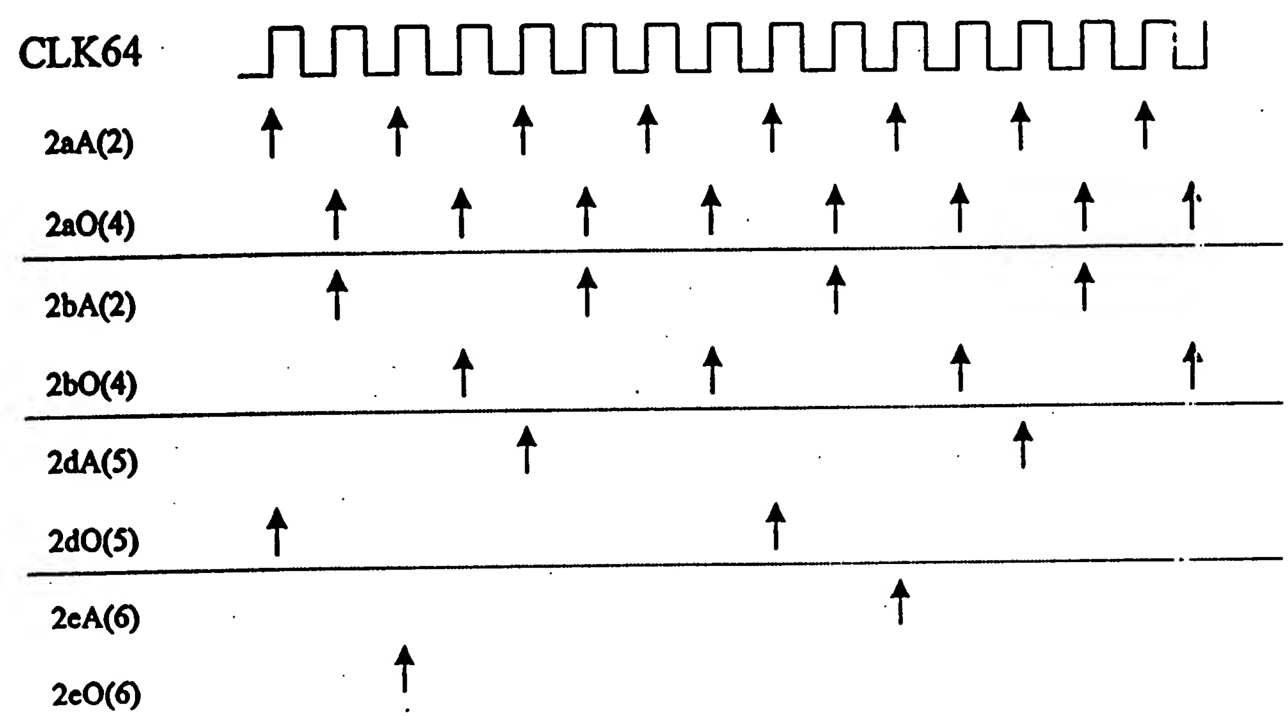


Figure 18A
Figure 18B
Figure 18C
Figure 18D
Figure 18E
Figure 18F
Figure 18G
Figure 18H
Figure 18I

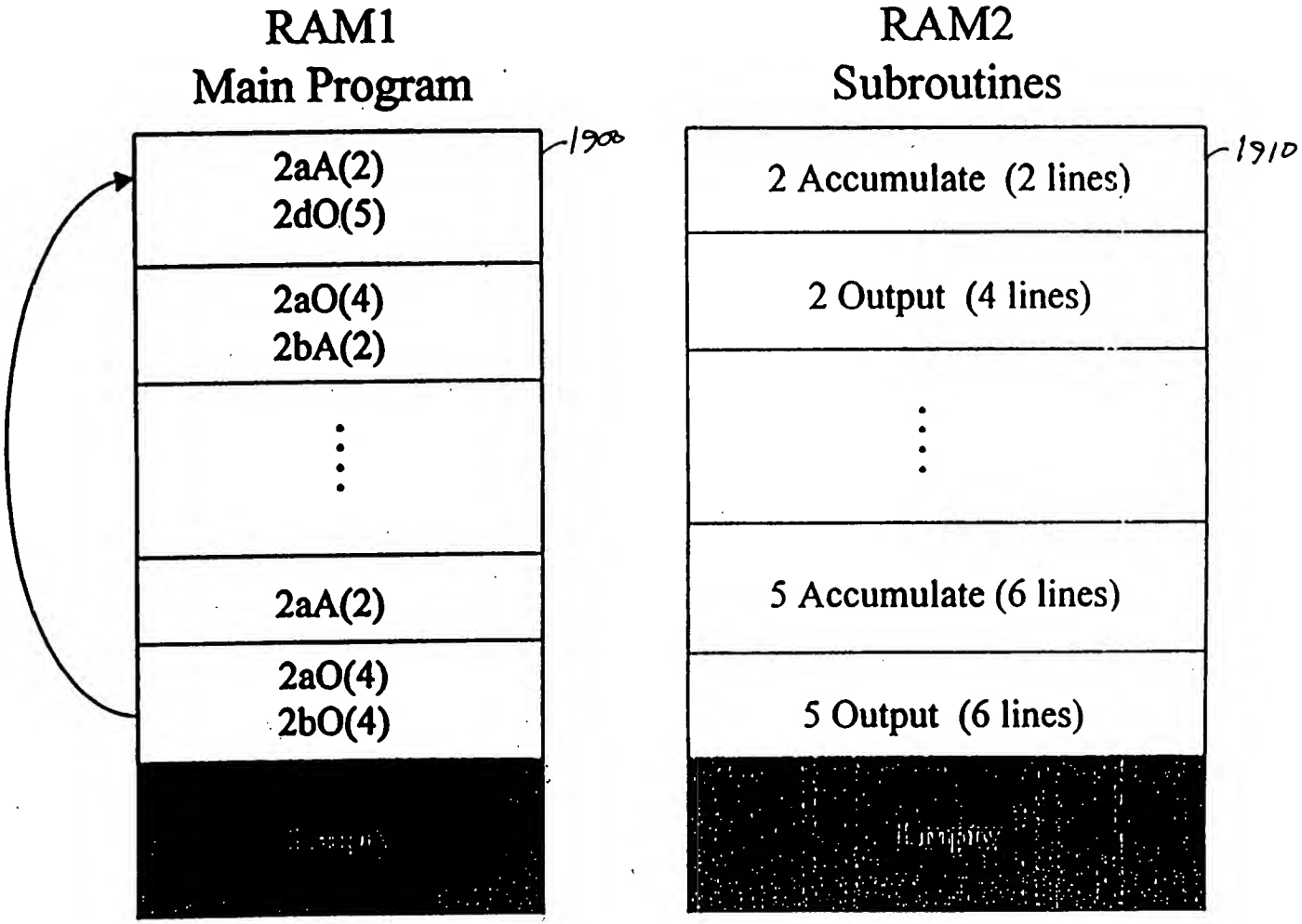


FIGURE 19A

FIGURE 19B

Sinc2 Control-Datapath Architecture

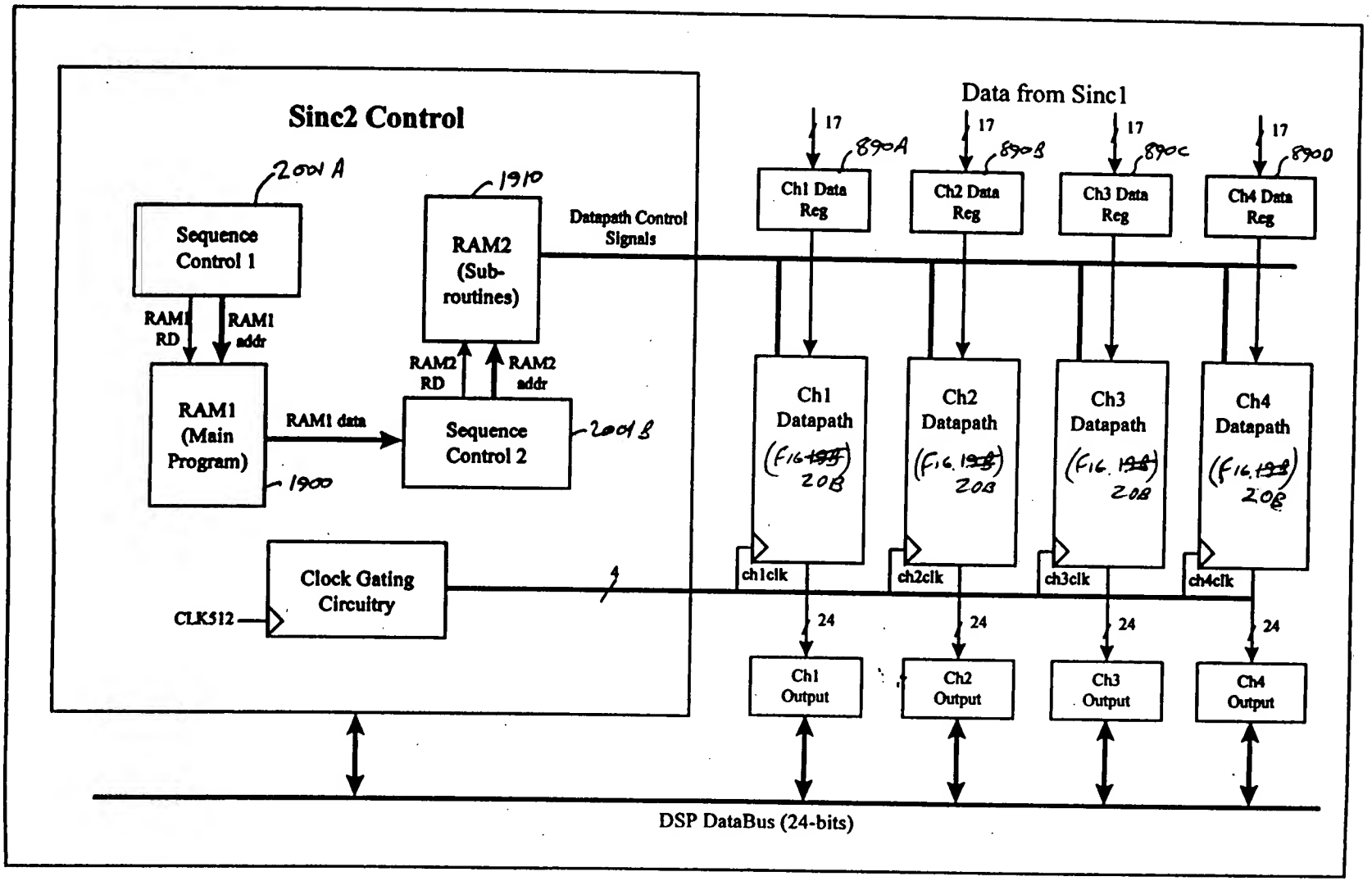


FIGURE 20A

09839945.030801

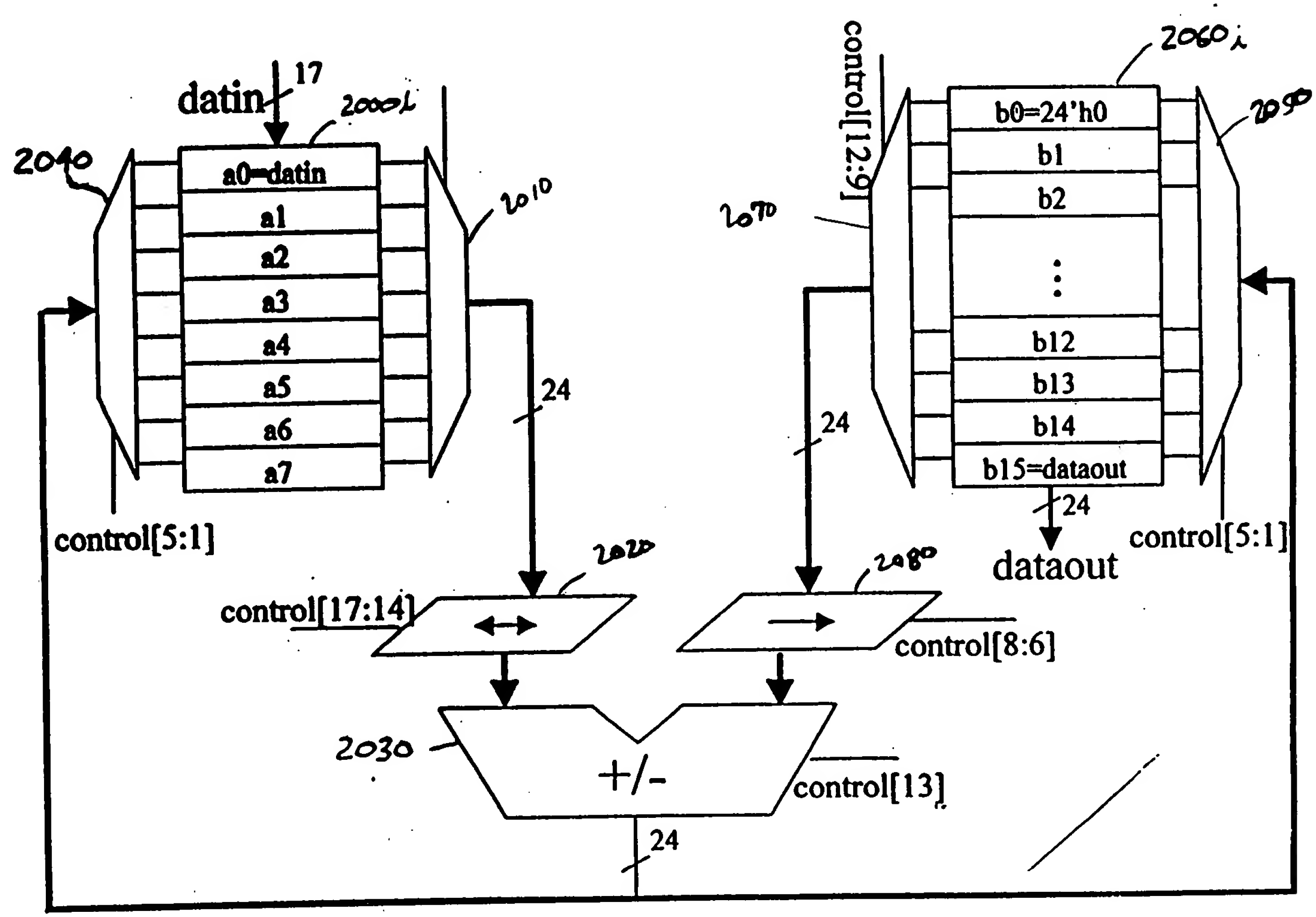


FIGURE 20B

Programming Procedure:

1. Select decimation rate.

2. Select required mini-sincs and associated Accumulate and Output subroutines.

3. Separate coefficients into form suitable for shift-add operations.

4. Check for overflow after each addition in the filter.

5. Perform necessary truncation to 24 bits and scaling of subsequent coefficients in mini-sincs.

6. Time multiplex Accumulate and Output Subroutines so that a maximum of 8 additions/subtractions are performed for each input from sinc1.

7. Create code for RAM2 (Accumulate and Output Subroutines) in the form:
   [Coeff 1] [Src 1] [Src 2] [Dest] [Coeff2] [Done Subroutine]

8. Create code for RAM1 (Main Control code)
   [Line #] [Wait for new data] [Done program]

FIGURE 21

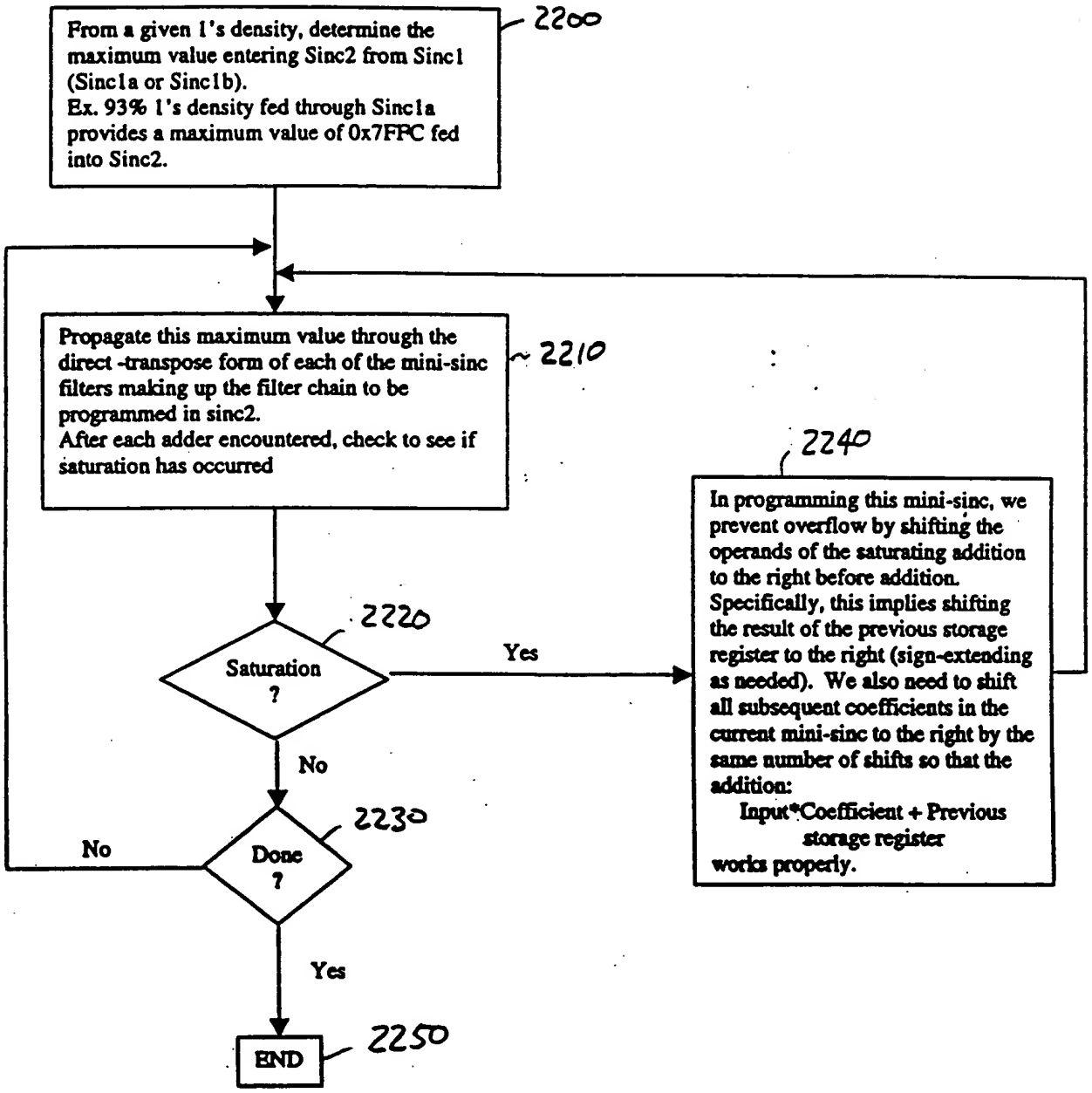

From a given 1's density, determine the maximum value entering Sinc2 from Sinc1 (Sinc1a or Sinc1b).
Ex. 93% 1's density fed through Sinc1a provides a maximum value of 0x7FPC fed into Sinc2.
2200
Propagate this maximum value through the direct -transpose form of each of the mini-sinc filters making up the filter chain to be programmed in sinc2.
After each adder encountered, check to see if saturation has occurred
2210
2240
In programming this mini-sinc, we prevent overflow by shifting the operands of the saturating addition to the right before addition. Specifically, this implies shifting the result of the previous storage register to the right (sign-extending as needed). We also need to shift all subsequent coefficients in the current mini-sinc to the right by the same number of shifts so that the addition:
Input*Coefficient + Previous storage register
works properly.
2220
Saturation ?
Yes
No
2230
Done ?
No
Yes
END
2250

FIGURE 22

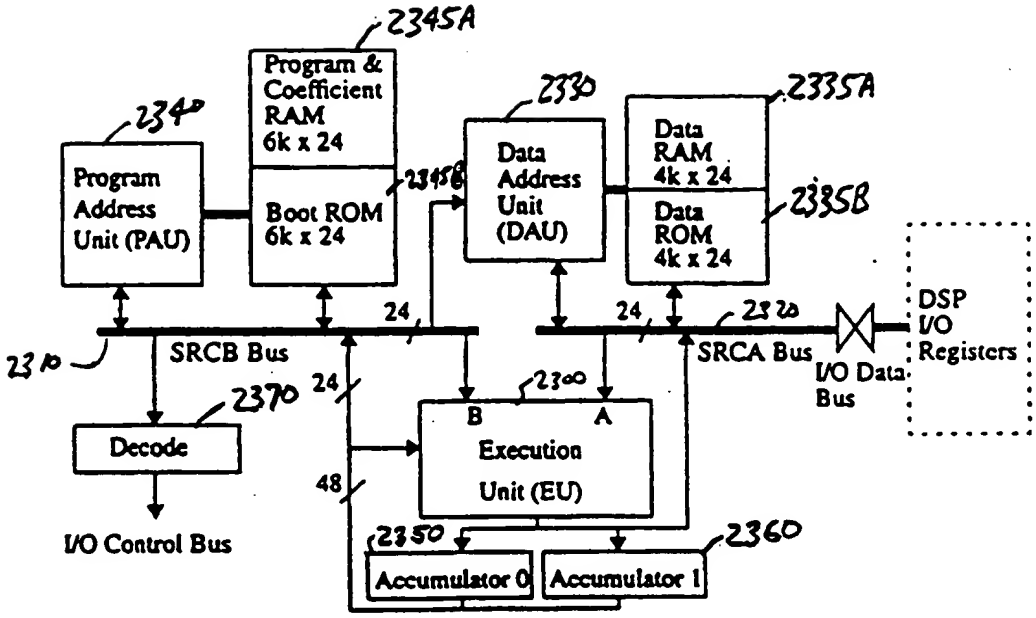

2345A
Program & Coefficient RAM 6k x 24
2340
Program Address Unit (PAU)
Boot ROM 6k x 24
2345B
2330
Data Address Unit (DAU)
2335A
Data RAM 4k x 24
2335B
Data ROM 4k x 24
24
24
2320
DSP I/O Registers
I/O Data Bus
SRCA Bus
2310
SRCB Bus
2370
Decode
I/O Control Bus
24
2300
B
A
Execution Unit (EU)
48
2350
2360
Accumulator 0
Accumulator 1

FIGURE 23

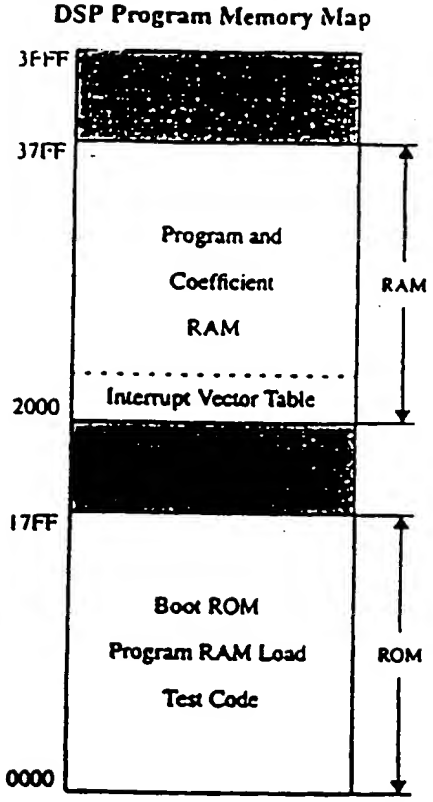


FIGURE 24A

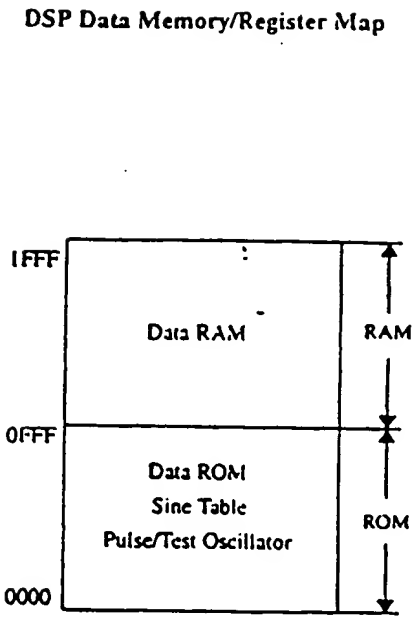


FIGURE 24B

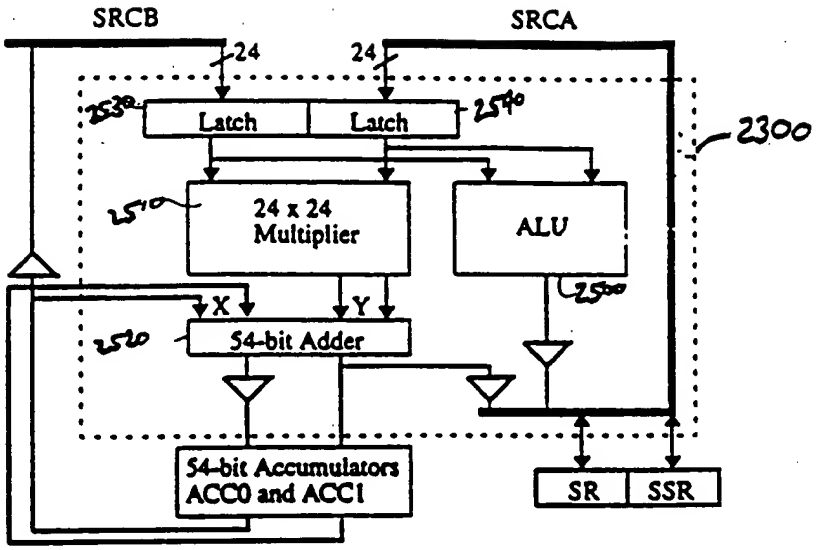

SRCB
SRCA
24
24
Latch
Latch
2300
24 x 24
Multiplier
ALU
X
Y
54-bit Adder
54-bit Accumulators
ACC0 and ACC1
SR
SSR

FIGURE 25

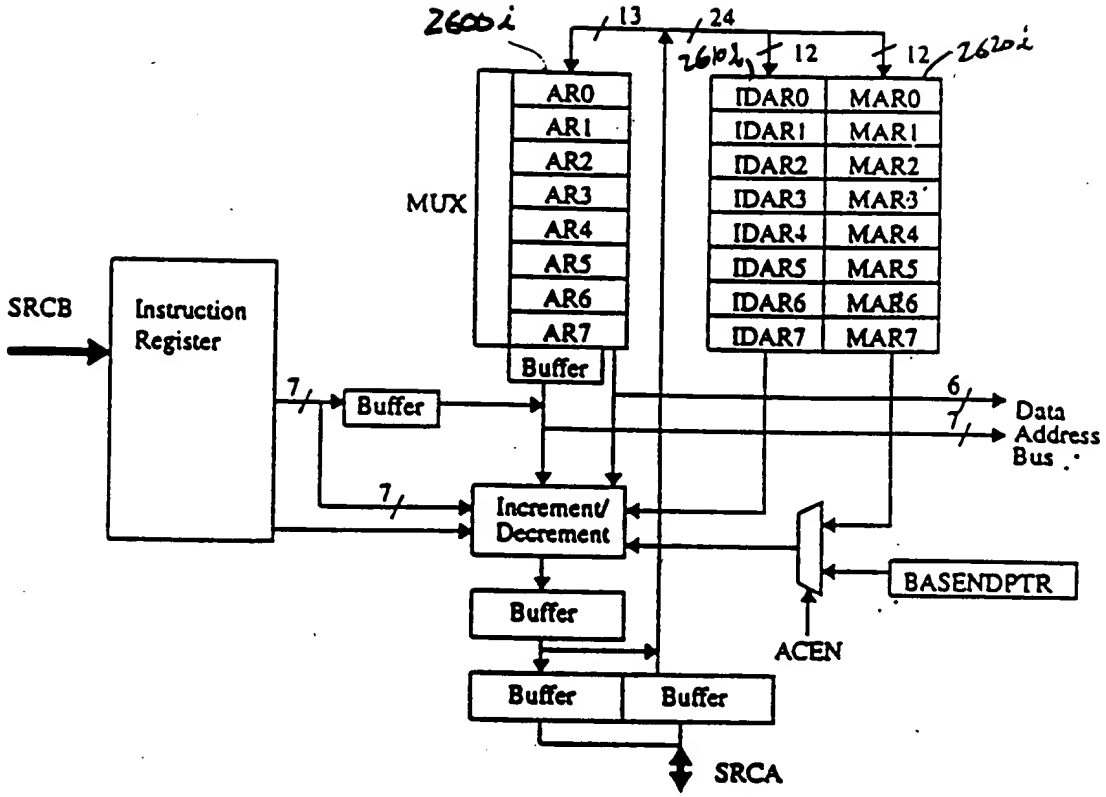

2600i
13
24
2602i
12
12
2620i
AR0
AR1
AR2
AR3
AR4
AR5
AR6
AR7
MUX
IDAR0
IDAR1
IDAR2
IDAR3
IDAR4
IDAR5
IDAR6
IDAR7
MAR0
MAR1
MAR2
MAR3
MAR4
MAR5
MAR6
MAR7
SRCB
Instruction Register
Buffer
7
Buffer
6
7
Data Address Bus
7
Increment/ Decrement
BASENDPTR
Buffer
ACEN
Buffer
Buffer
SRCA

FIGURE 26

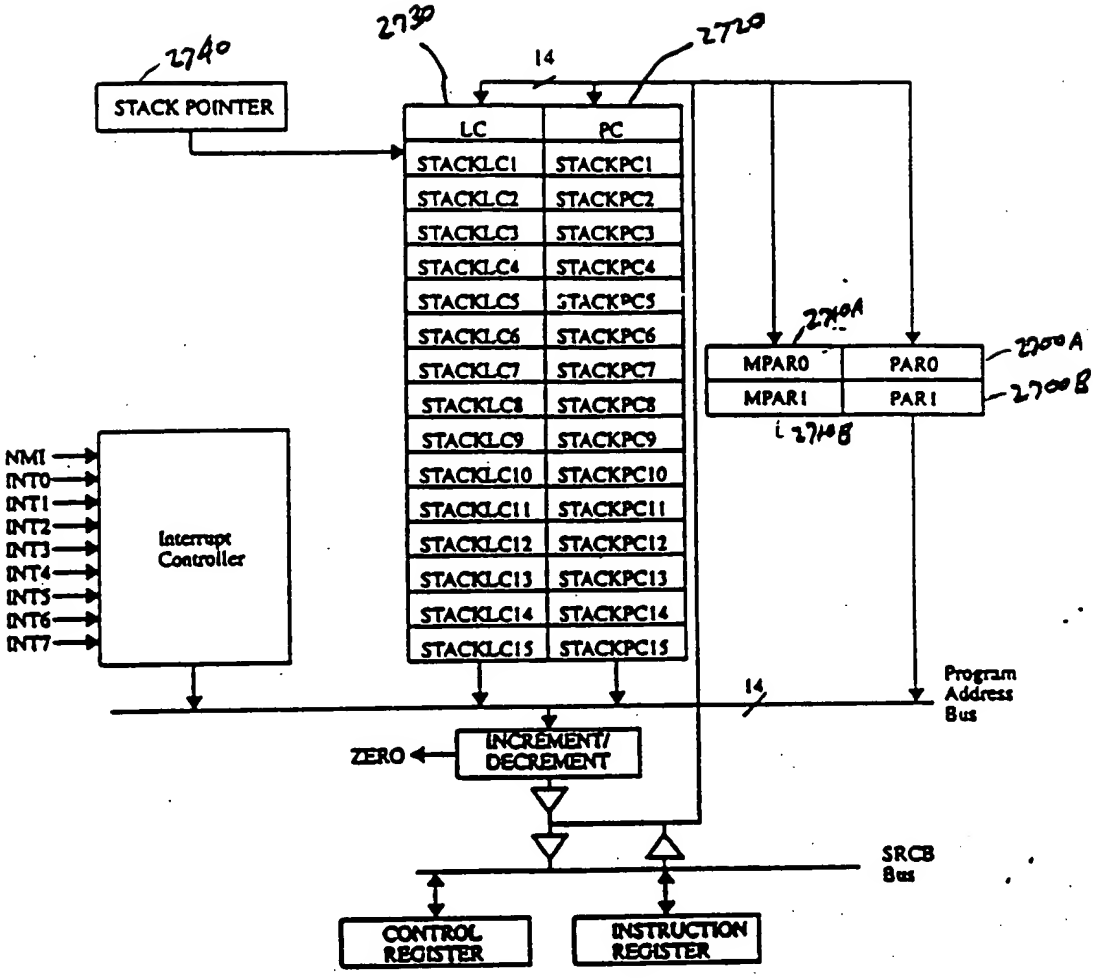

2740
STACK POINTER
2730
14
2720
LC
PC
STACKLC1
STACKPC1
STACKLC2
STACKPC2
STACKLC3
STACKPC3
STACKLC4
STACKPC4
STACKLC5
STACKPC5
STACKLC6
STACKPC6
STACKLC7
STACKPC7
STACKLC8
STACKPC8
STACKLC9
STACKPC9
STACKLC10
STACKPC10
STACKLC11
STACKPC11
STACKLC12
STACKPC12
STACKLC13
STACKPC13
STACKLC14
STACKPC14
STACKLC15
STACKPC15
2710A
MPAR0
PAR0
2700A
MPAR1
PAR1
2700B
2710B
NMI
INT0
INT1
INT2
INT3
INT4
INT5
INT6
INT7
Interrupt Controller
14
Program Address Bus
ZERO
INCREMENT/ DECREMENT
SRCB Bus
CONTROL REGISTER
INSTRUCTION REGISTER

FIGURE 27

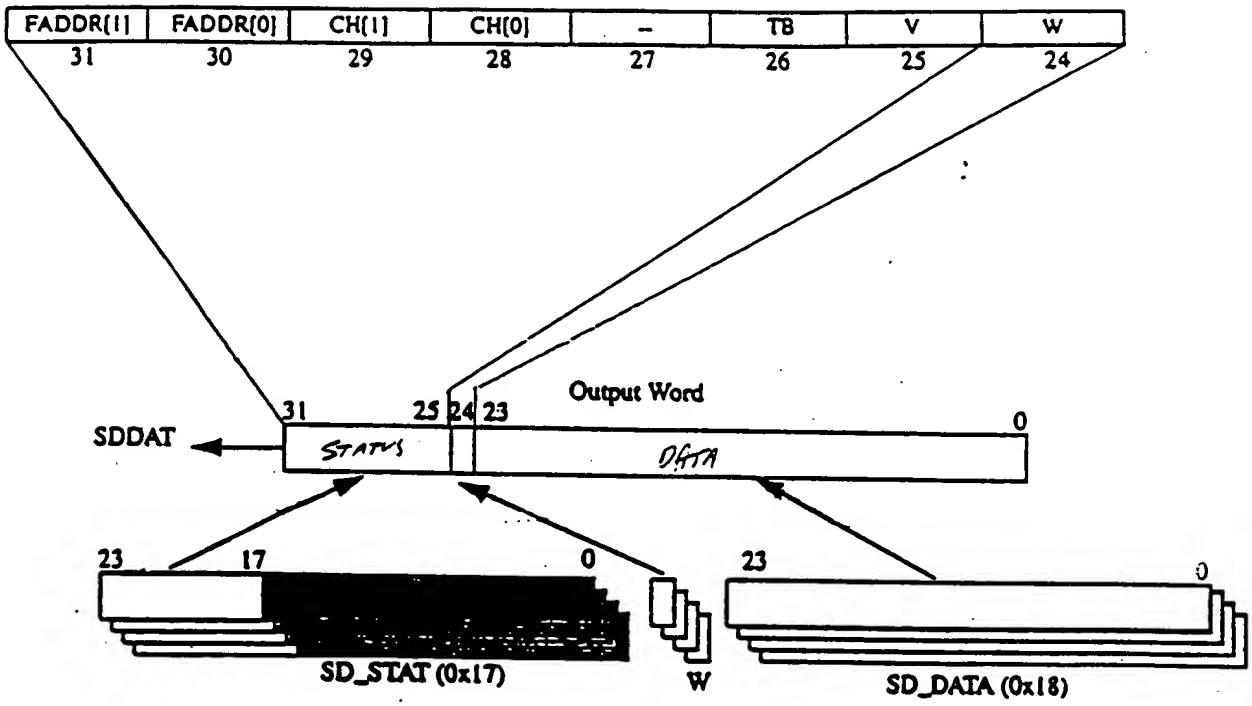

FADDR[1]
FADDR[0]
CH[1]
CH[0]
–
TB
V
W
31
30
29
28
27
26
25
24
Output Word
SDDAT
31
25
24
23
0
STATUS
DATA
23
17
0
SD_STAT (0x17)
W
23
0
SD_DATA (0x18)

FIGURE 28

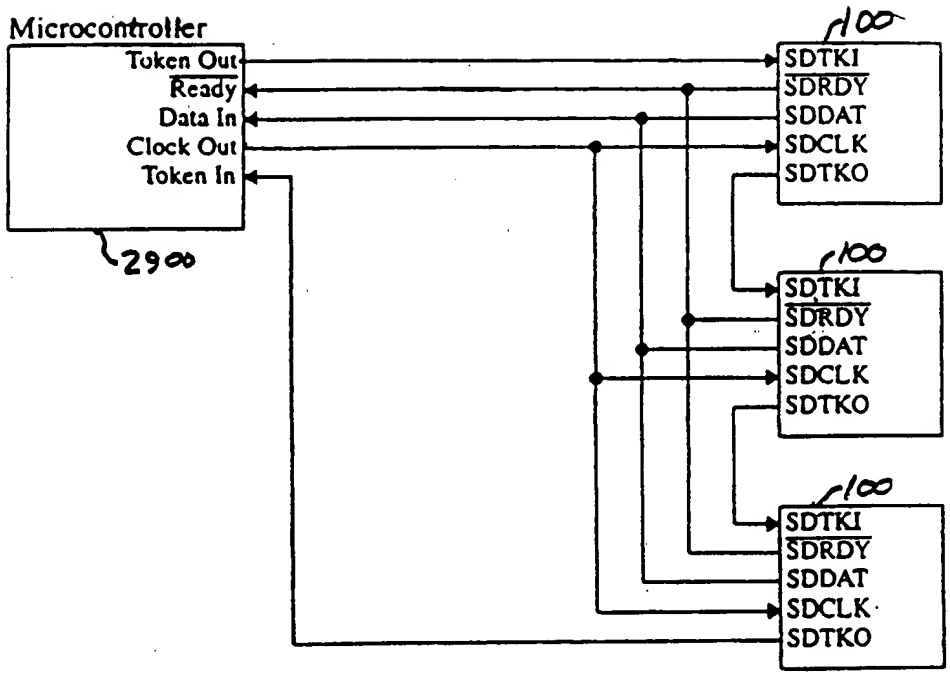

Microcontroller
Token Out
Ready
Data In
Clock Out
Token In
2900
100
SDTKI
SDRDY
SDDAT
SDCLK
SDTKO
100
SDTKI
SDRDY
SDDAT
SDCLK
SDTKO
100
SDTKI
SDRDY
SDDAT
SDCLK
SDTKO

FIGURE 29

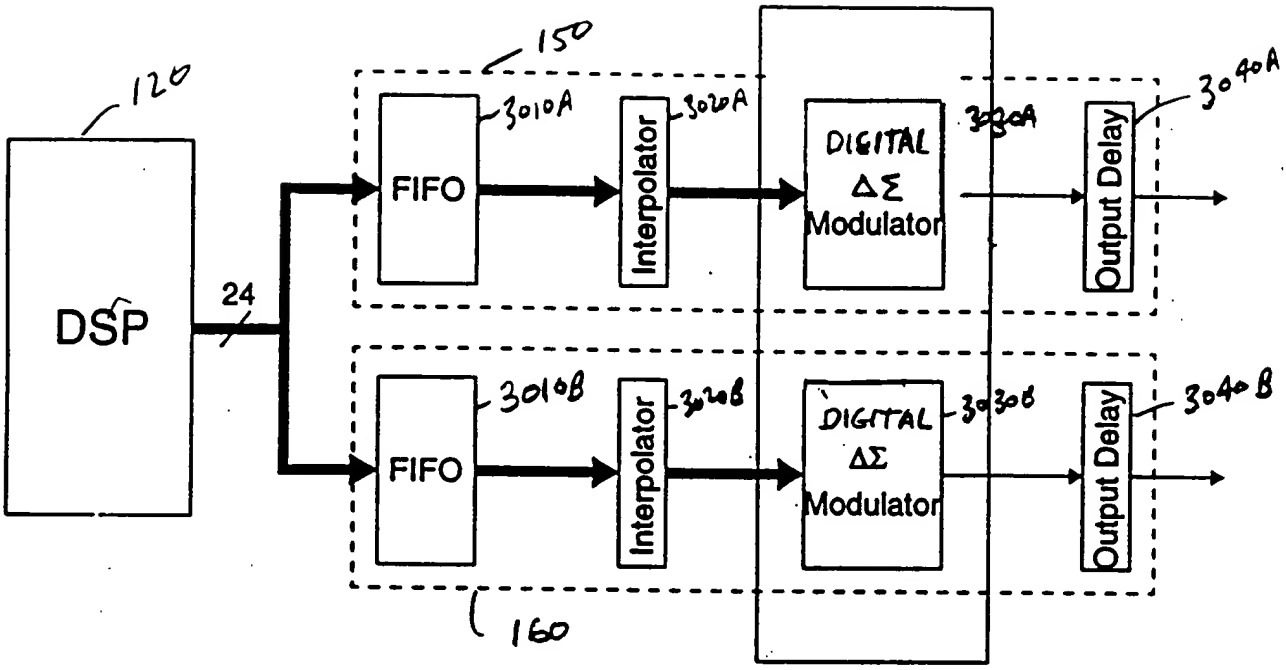

120
150
DSP
24
FIFO
3010A
Interpolator
3020A
DIGITAL
ΔΣ
Modulator
3030A
Output Delay
3040A
FIFO
3010B
Interpolator
3020B
DIGITAL
ΔΣ
Modulator
3030B
Output Delay
3040B
160

FIGURE 30A

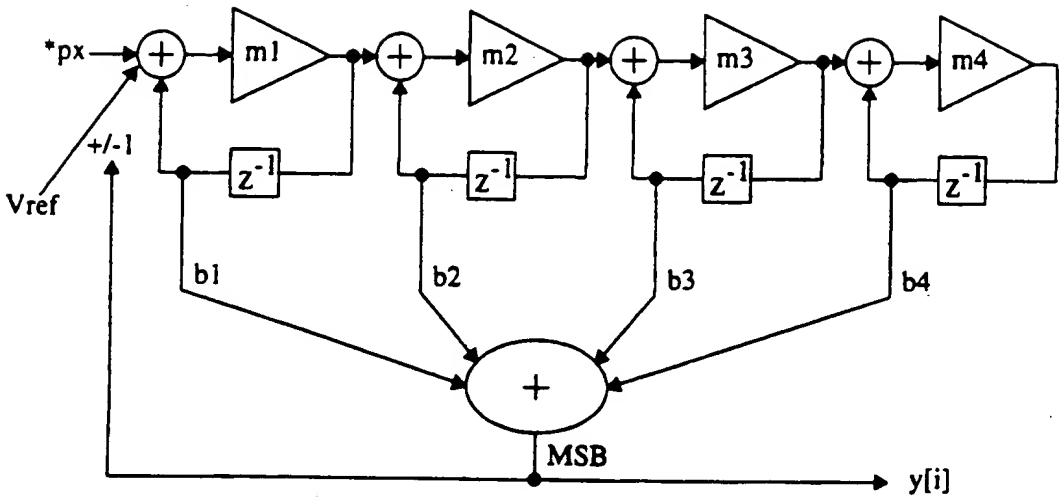
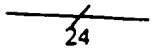


FIGURE 30B

FIGURE 30C1 — wire

FIGURE 30C2 — 

24 wires

FIGURE 30C3 —  register

FIGURE 30C4 —  multiplexer

FIGURE 30C5 —  tristate buffer

FIGURE 30C6 —  inverter

FIGURE 30C7 —  exclusive or gate

FIGURE 30C8 — 

adder

FIGURE 30C9 — 

multiplier

FIGURE 30C10 —  right shifter

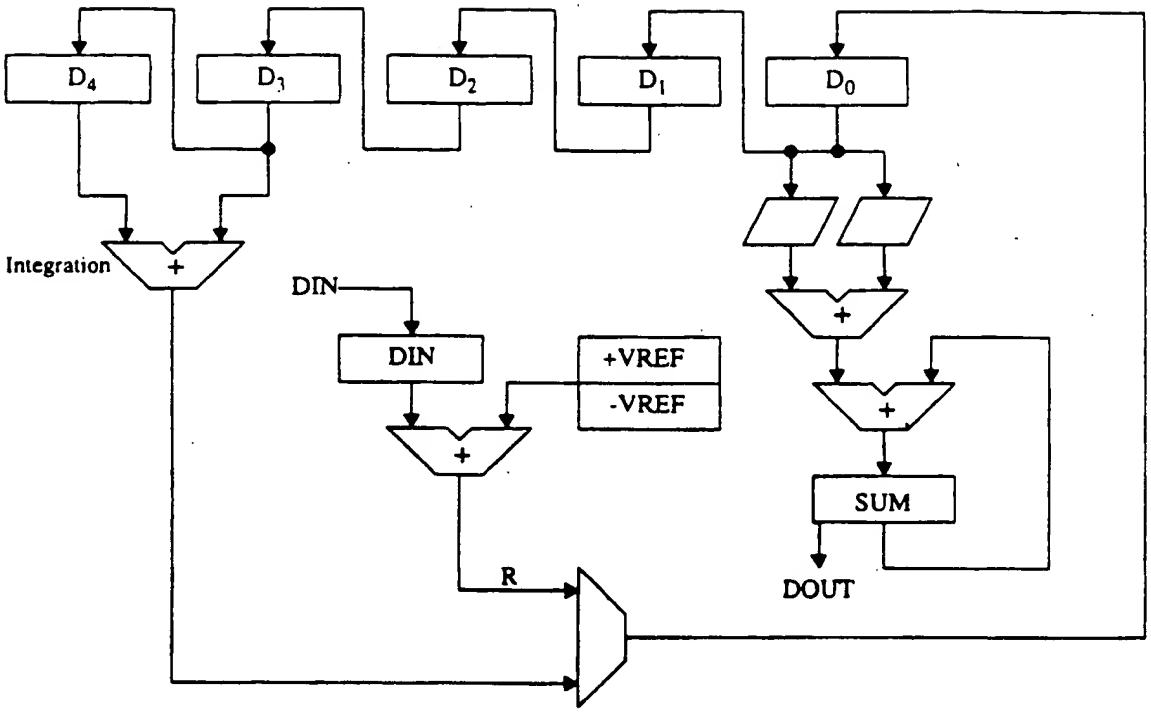


FIGURE 3OF1

State	Actions During State		
S0	$D_0(D4_k) = D_4(D4_{k-1}) + D_3(D3_{k-1})$	Clear SUM	Load DIN_k
S1	$D_0(D3_k) = D_4(D3_{k-1}) + D_3(D2_{k-1})$	SUM_k += $D_0(D4_k)$>>Shift4	
S2	$D_0(D2_k) = D_4(D2_{k-1}) + D_3(D1_{k-1})$	SUM_k += $D_0(D3_k)$>>Shift3	
S3	$D_0(D1_k) = D_4(D1_{k-1}) + D_3(R_{k-1})$	SUM_k += $D_0(D2_k)$>>Shift2	
S4		SUM_k += $D_0(D1_k)$>>Shift1	
S5	$D_0(R_k) = DIN_k$ +/- VREF		

FIGURE 3OF2

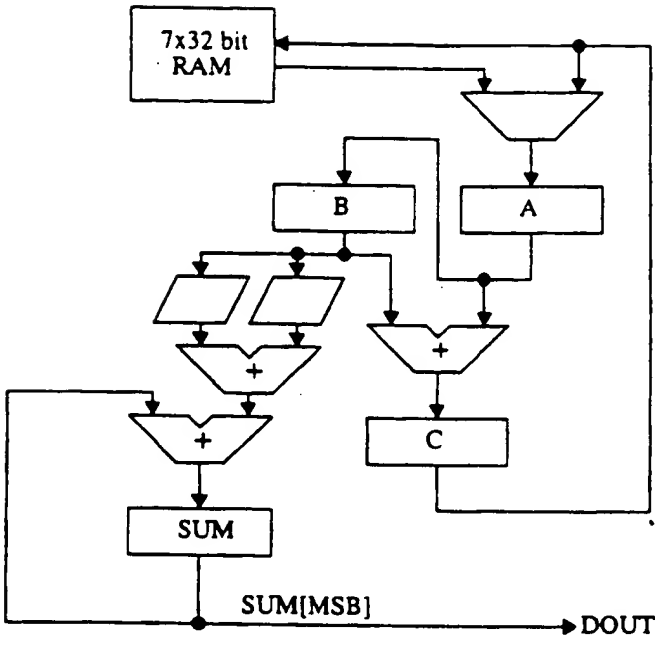


FIGURE 30G1

State	Actions During State			
S0	Clear SUM	Clear C	Clear B	Clear A
S1				Load A<Mem($D4_k$)
S2			Shift B<A($D4_k$)	Load A<Mem($D3_k$)
S3	SUM_k += B($D4_k$)>>Shift4	C = B($D4_k$) + A($D3_k$)	Shift B<A($D3_k$)	Load A<Mem($D2_k$)
S4				Store C>Mem($D4_{k+1}$)
S5	SUM_k += B($D3_k$)>>Shift3	C = B($D3_k$) + A($D2_k$)	Shift B<A($D2_k$)	Load A<Mem($D1_k$)
S6				Store C>Mem($D3_{k+1}$)
S7	SUM_k += B($D2_k$)>>Shift2	C = B($D2_k$) + A($D1_k$)	Shift B<A($D1_k$)	Load A<Mem(DIN_k)
S8				Store C>Mem($D2_{k+1}$)
S9	SUM_k += B($D1_k$)>>Shift1	C = B($D1_k$) + A(DIN_k)	Shift B<A(DIN_k)	Load A<Mem(VREF)
S10			Shift B<A(VREF)	LoadReg A<C(Temp)
S11		C = +/-B(VREF) + A(Temp)		
S12				Store C>Mem($D1_{k+1}$)

FIGURE 30G2

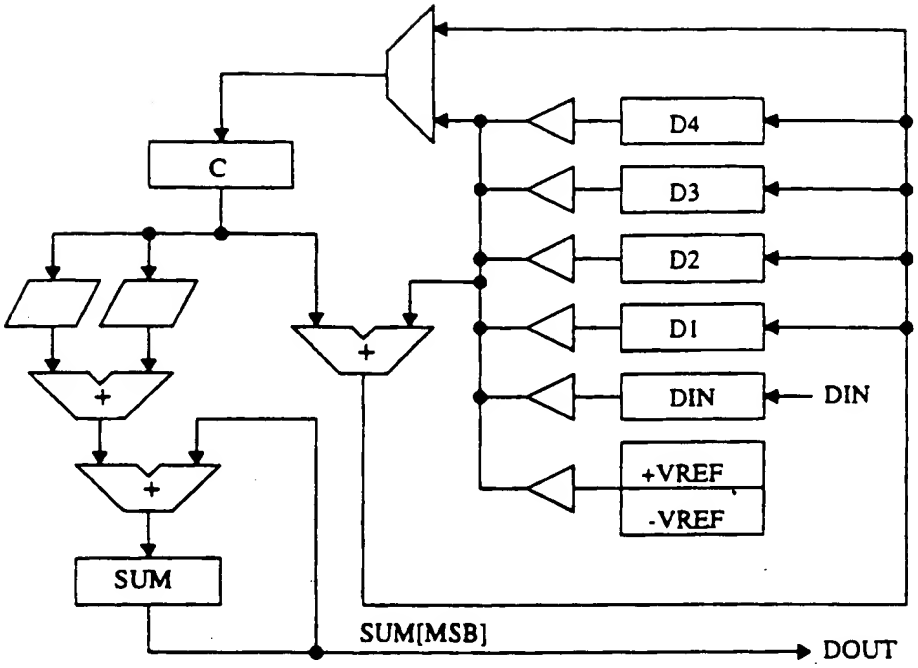


FIGURE 30 H1

State	Actions During State			
S0	Clear SUM	Load C < $D4_k$		Load DIN_k
S1	SUM_k += C($D4_k$)>>Shift4	Load C < $D3_k$	$D4_{k+1}$ = C($D4_k$) + $D3_k$	
S2	SUM_k += C($D3_k$)>>Shift3	Load C < $D2_k$	$D3_{k+1}$ = C($D3_k$) + $D2_k$	
S3	SUM_k += C($D2_k$)>>Shift2	Load C < $D1_k$	$D2_{k+1}$ = C($D2_k$) + $D1_k$	
S4	SUM_k += C($D1_k$)>>Shift1	C(Temp) = C($D1_k$) + DIN_k		
S5			$D1_{k+1}$ = C(Temp) +/- VREF	

FIGURE 30H2

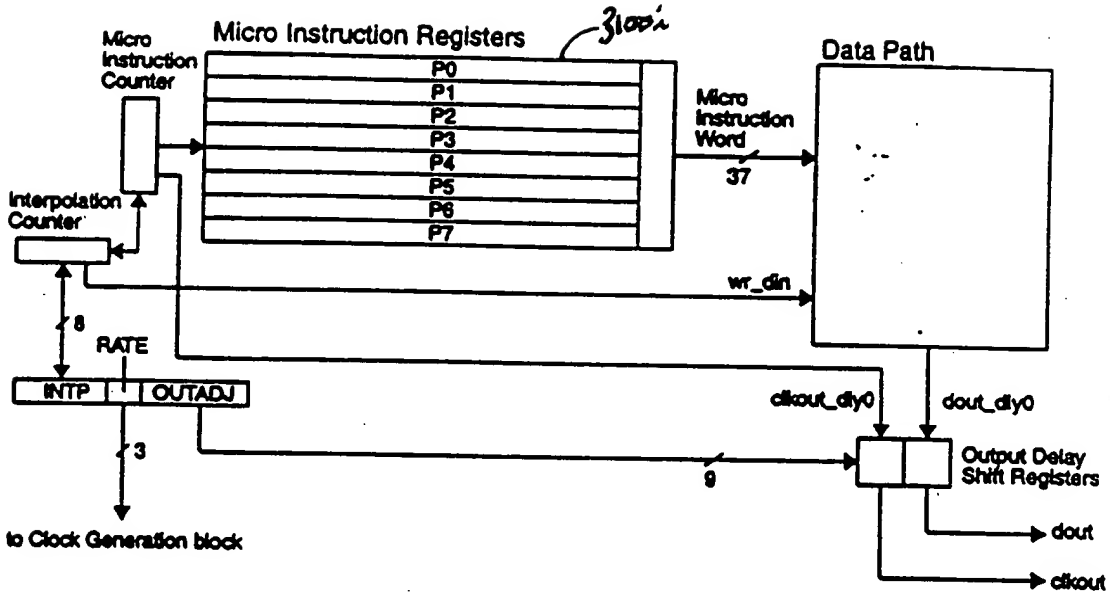

Micro Instruction Registers
Micro Instruction Counter
P0
P1
P2
P3
P4
P5
P6
P7
Micro Instruction Word
37
Data Path
Interpolation Counter
wr_din
8
RATE
INTP
OUTADJ
3
to Clock Generation block
clkout_dly0
dout_dly0
9
Output Delay Shift Registers
dout
clkout

FIGURE 31

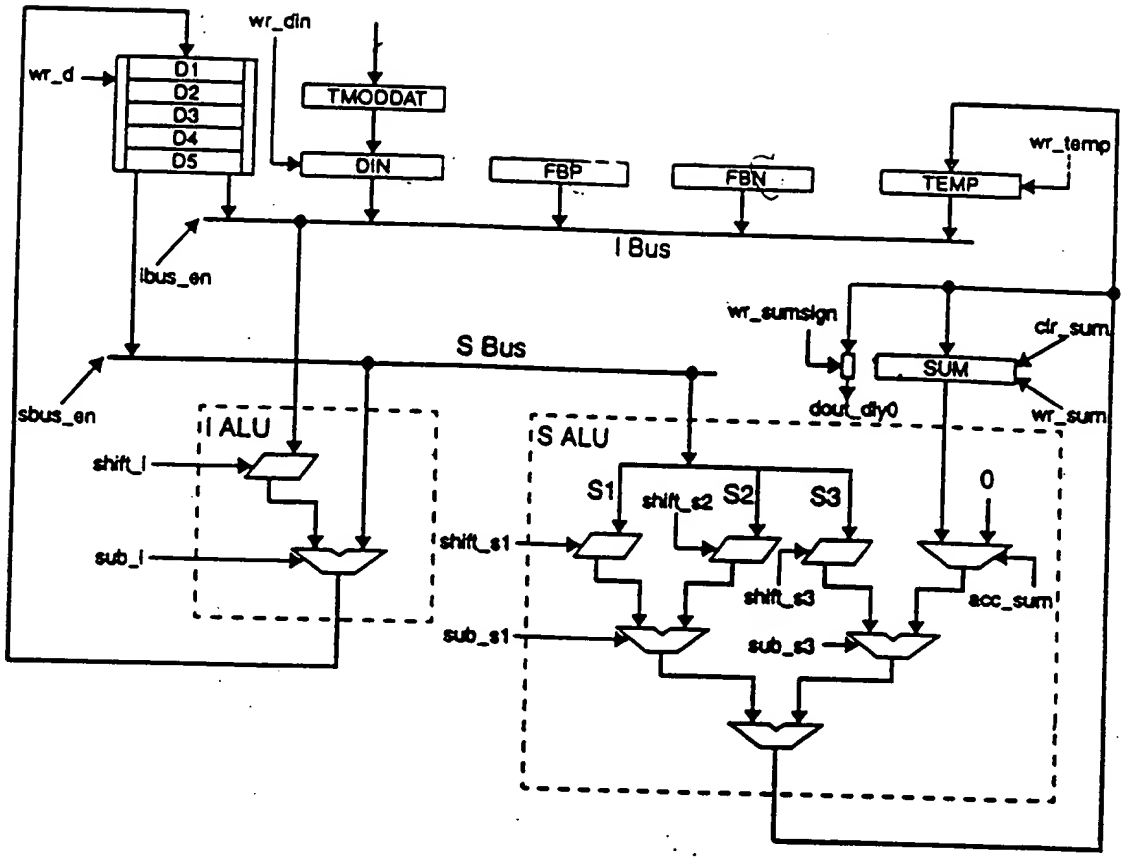

wr_din
wr_d
D1
D2
D3
D4
D5
TMODDAT
DIN
FBP
FBN
TEMP
wr_temp
I Bus
ibus_en
wr_sumsign
clr_sum
S Bus
SUM
sbus_en
dout_dly0
wr_sum
I ALU
S ALU
shift_I
sub_I
S1
shift_s2
S2
S3
0
shift_s1
shift_s3
acc_sum
sub_s1
sub_s3

FIGURE 32

P	Feedforward	Integration	Temp	Din	SUM	SUMSIGN	TEMP	S Bus	I Bus	Write I
0	$SUM_k = D4_k>>11 + D4_k>>9 + D4_k>>7$	$D4_{k+1} = D4_k + D3_k$		Load DIN_k	Write			+D4>>7 +D4>>9 +D4>>11	+D3	D4
1	$SUM_k = SUM_k + D3_k>>8 + D3_k>>5 + D3_k>>4$	$D3_{k+1} = D3_k + D2_k$			Acc./ Write			+D3>>4 +D3>>5 +D3>>8	+D2	D3
2	$SUM_k = SUM_k + D2_k>>1 - D2_k>>7 - D2_k>>4$	$D2_{k+1} = D2_k + D1_k$			Acc./ Write			-D2>>4 +D2>>1 -D2>>7	+D1	D2
3	$SUM_k = SUM_k + D1_k$	$D1_{k+1}' = D1_k + DIN_k$			Acc./ Write	Write		+D1 +D1 -D1	+DIN	D1
4		$D1_{k+1} = D1_{k+1}' +/- VREF$							+FB	D1
5										
6										
7										

FIGURE 33

G	C	S	T	wr_d			ibus_en			sbus_en			shift_i					s1	shift_s1					ss1	shift_s2					shift_s3					ss3	A	HEX
36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	1	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0	1	0	1	0	1	1	0	0	051C 00E4AC
0	0	1	0	0	1	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	1	0	0	0	0	1	04D3 0082A1
0	0	1	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	1	1	1	1	1	048A 00909F
1	0	1	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1469 000003
0	0	0	0	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0079 000000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 000000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 000000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 000000

FIGURE 34

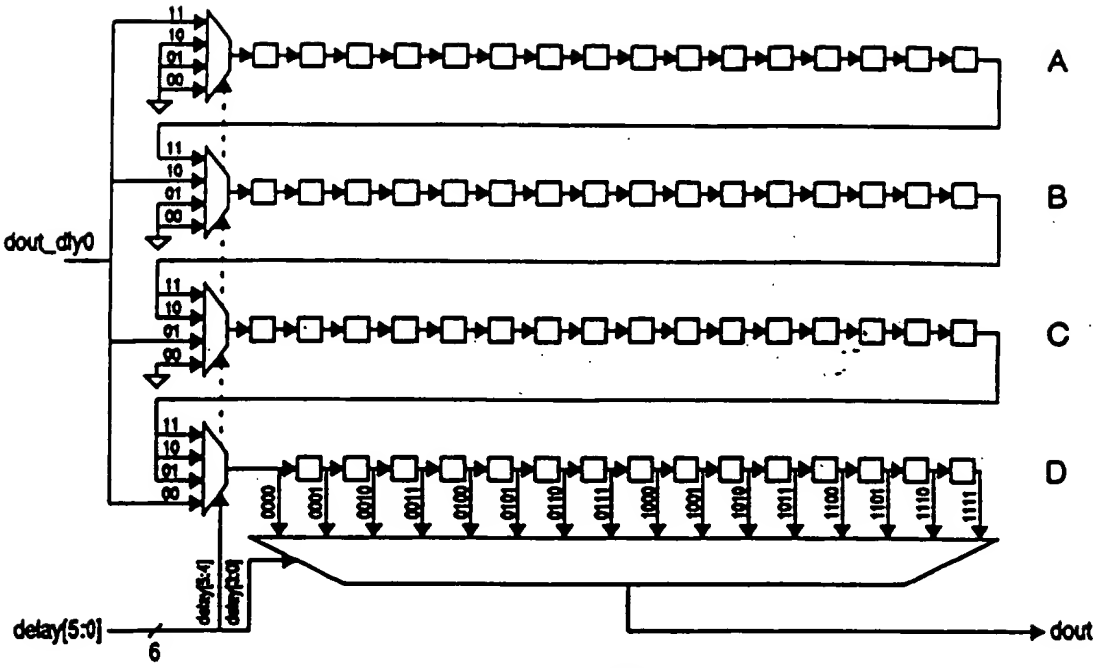


FIGURE 35A

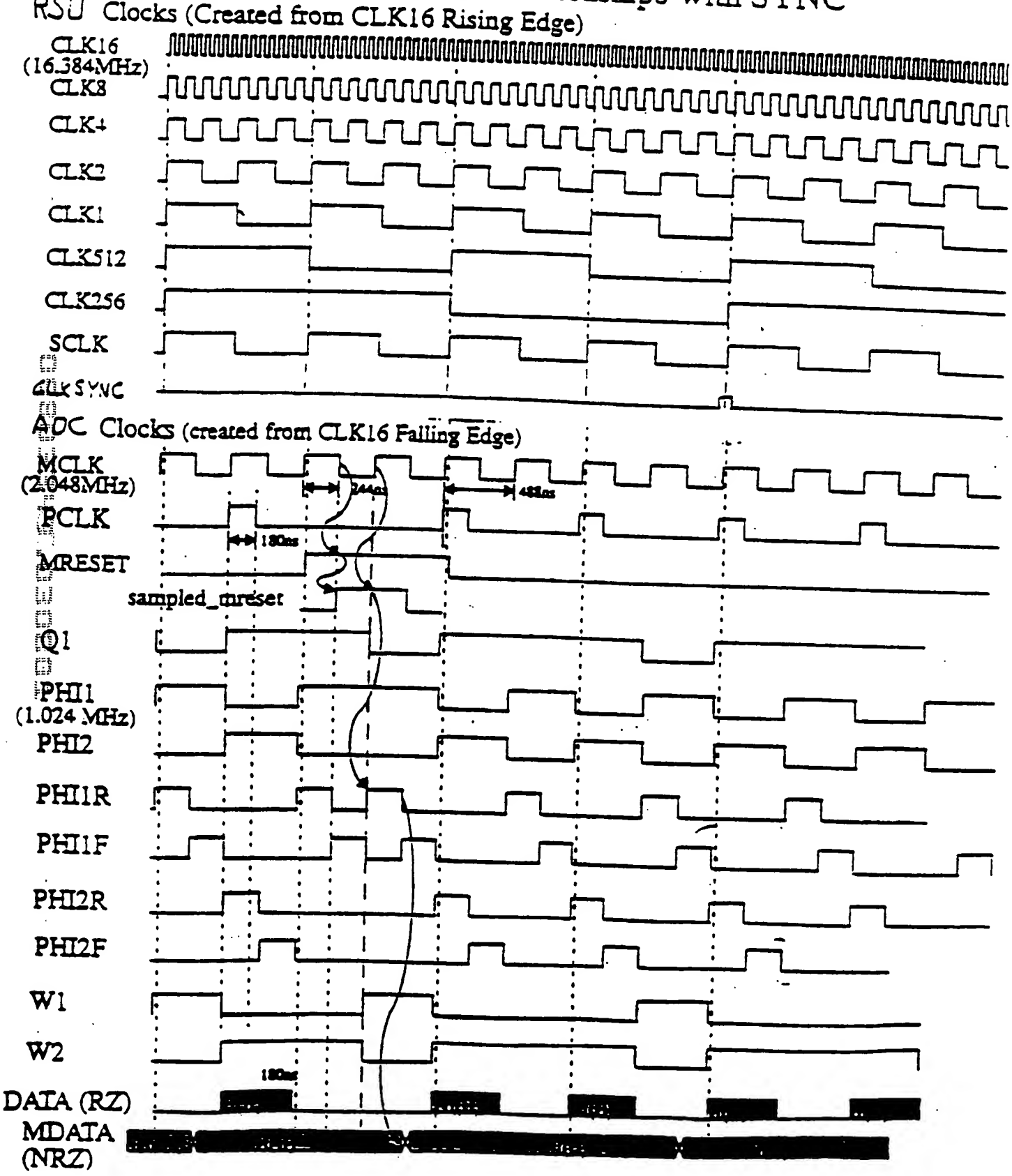
Table 1: Legend

dout_dly0	data output bit, 0 delay
dout	data output bit, 0-63 clock delay
delay[5:0]	how many clocks (0-63) to delay output data dout_dly0
delay[5:4]	selects segment into which to direct dout_dly0
delay[3:0]	selects where to tap segment D to get dout

FIGURE 35B

FIGURE 36

RSU & ADC interface Clock Relationships with SYNC



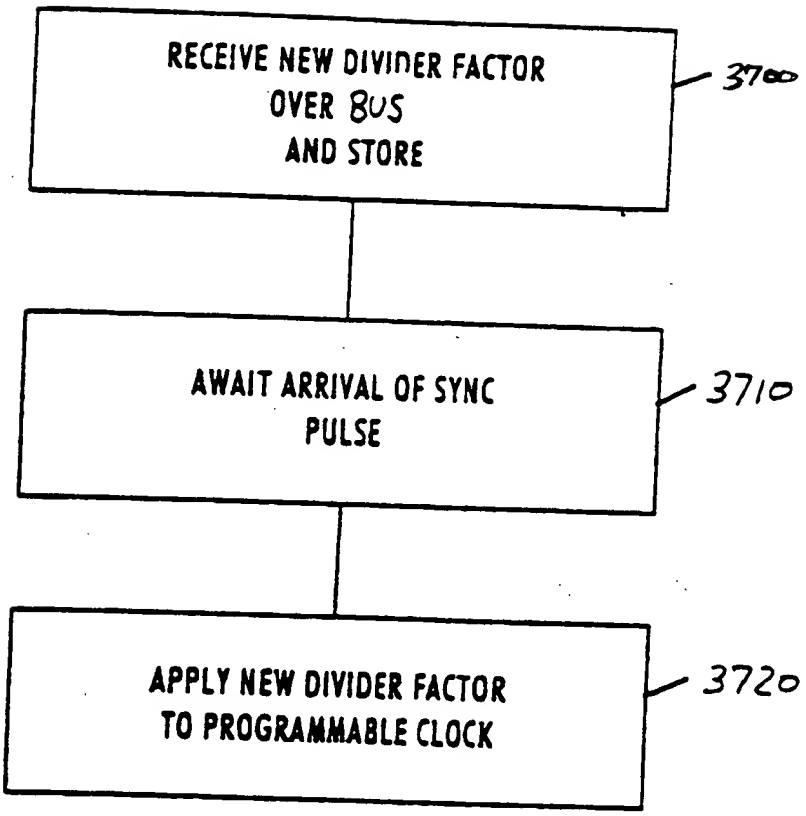

RECEIVE NEW DIVIDER FACTOR OVER BUS AND STORE
3700
AWAIT ARRIVAL OF SYNC PULSE
3710
APPLY NEW DIVIDER FACTOR TO PROGRAMMABLE CLOCK
3720

Figure 37